arm

The Salmon Diet

Up-streaming Device Drivers as a Form of Optimization

Embedded Linux Conference

Gilad Ben-Yossef
Principal Software Engineer

About me

My name is Gilad Ben-Yossef.

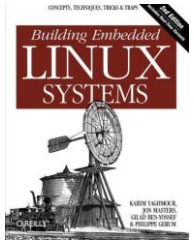
I work on upstream Linux kernel cryptography and security and the maintainer of the arm® TrustZone® CryptoCell® device driver.

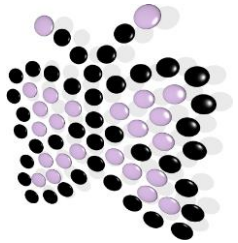
I've been working in various forms with and on the Linux kernel and other Open Source projects for close to twenty years – i.e. I'm an old bugger... ☺

I co-authored "Building Embedded Linux Systems" 2nd edition from O'Reilly.

I'm a co-founder of HaMakor, an Israeli NPO for free and open source software and of August Penguin, the longest running Israeli FOSS conference.



arm

arm

Salmon leaping at Willamette Falls

NOAA's Historic Fisheries Collection Location: Oregon, Oregon City

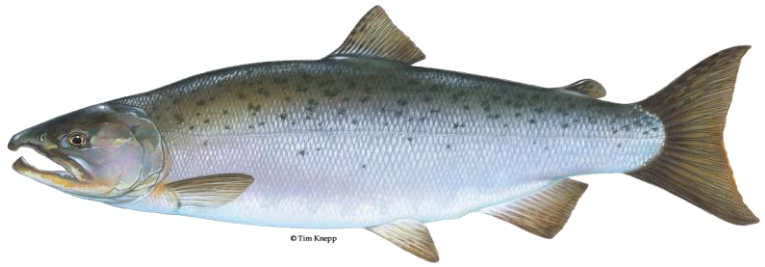
arm

Salmon leaping at the Linux staging tree

“The Linux Staging tree (or just "staging" from now on) is used to hold stand-alone drivers and filesystems that are not ready to be merged into the main portion of the Linux kernel tree at this point in time for various technical reasons.

It is contained within the main Linux kernel tree so that users can get access to the drivers much easier than before, and to provide a common place for the development to happen, resolving the "hundreds of different download sites" problem that most out-of-tree drivers have had in the past. “

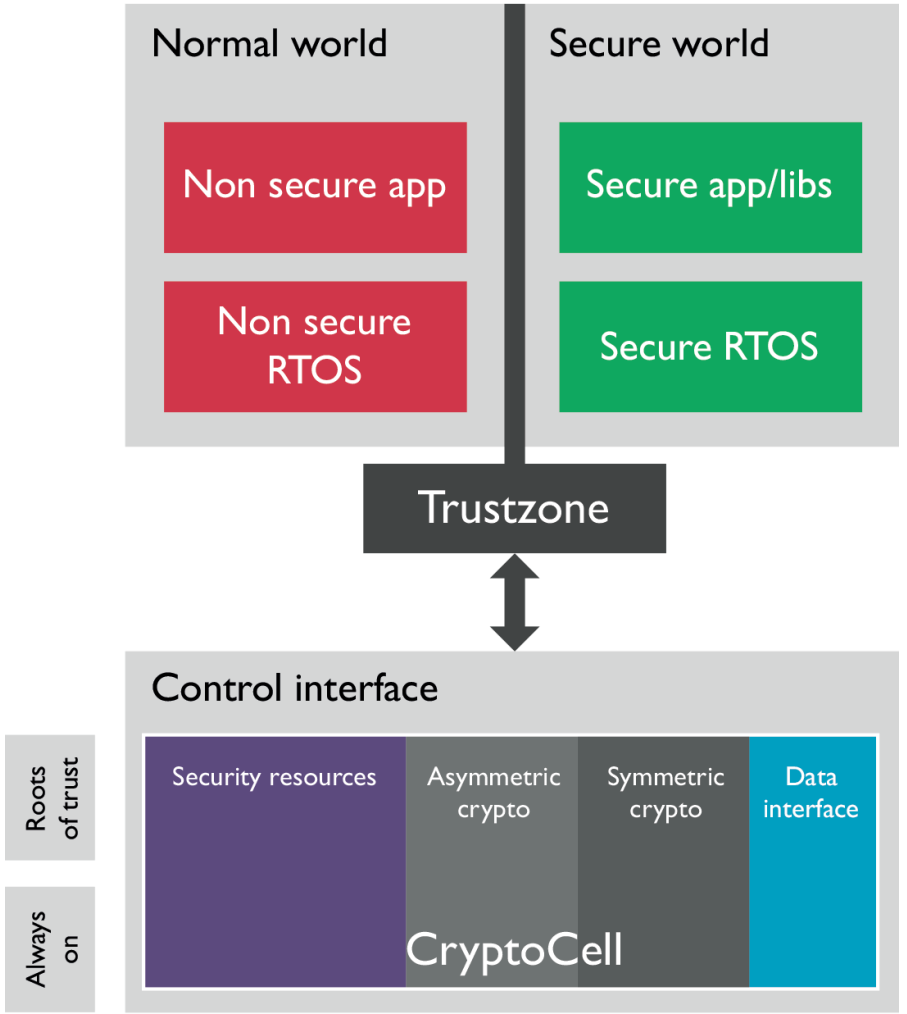
Greg KH



arm

What is Arm® TrustZone® CryptoCell?

- Readers Digest version:
- It's a HW block that handles system security.
- It provides crypto, root of trust, secure boot and debug.
- It goes in a SoC, outside the core.
- It serves both trusted and untrusted worlds.



arm

Taking upstream a long out-of-tree driver

```
commit abefd6741d540fc624e73a2a3bdef2397bcbd064
Author: Gilad Ben-Yossef <gilad@benyossef.com>
Date:   Sun Apr 23 12:26:09 2017 +0300

   staging: ccree: introduce CryptoCell HW driver

   Introduce basic low level Arm TrustZone CryptoCell HW support.
   This first patch doesn't actually register any Crypto API
   transformations, these will follow up in the next patch.

   This first revision supports the CC 712 REE component.

   Signed-off-by: Gilad Ben-Yossef <gilad@benyossef.com>
   Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
```

arm

Checkpatch Hell

CHECK: Alignment should match open parenthesis
#1485: FILE: drivers/staging/ccree/ssi_cipher.c:1485:
+ SSI_LOG_DEBUG("%s alg registration rc = %x\n",
+ t_alg->crypto_alg.cra_driver_name, rc);

CHECK: Alignment should match open parenthesis
#1488: FILE: drivers/staging/ccree/ssi_cipher.c:1488:
+ SSI_LOG_ERR("%s alg registration failed\n",
+ t_alg->crypto_alg.cra_driver_name);

ERROR: trailing whitespace
#1492: FILE: drivers/staging/ccree/ssi_cipher.c:1492:
+^I^I^Ilist_add_tail(&t_alg->entry, $

ERROR: trailing whitespace
#1494: FILE: drivers/staging/ccree/ssi_cipher.c:1494:
+^I^I^ISSI_LOG_DEBUG("Registered %s\n", $

CHECK: Alignment should match open parenthesis
#1495: FILE: drivers/staging/ccree/ssi_cipher.c:1495:
+ SSI_LOG_DEBUG("Registered %s\n",
+ t_alg->crypto_alg.cra_driver_name);

total: 134 errors, 112 warnings, 87 checks, 1503 lines checked

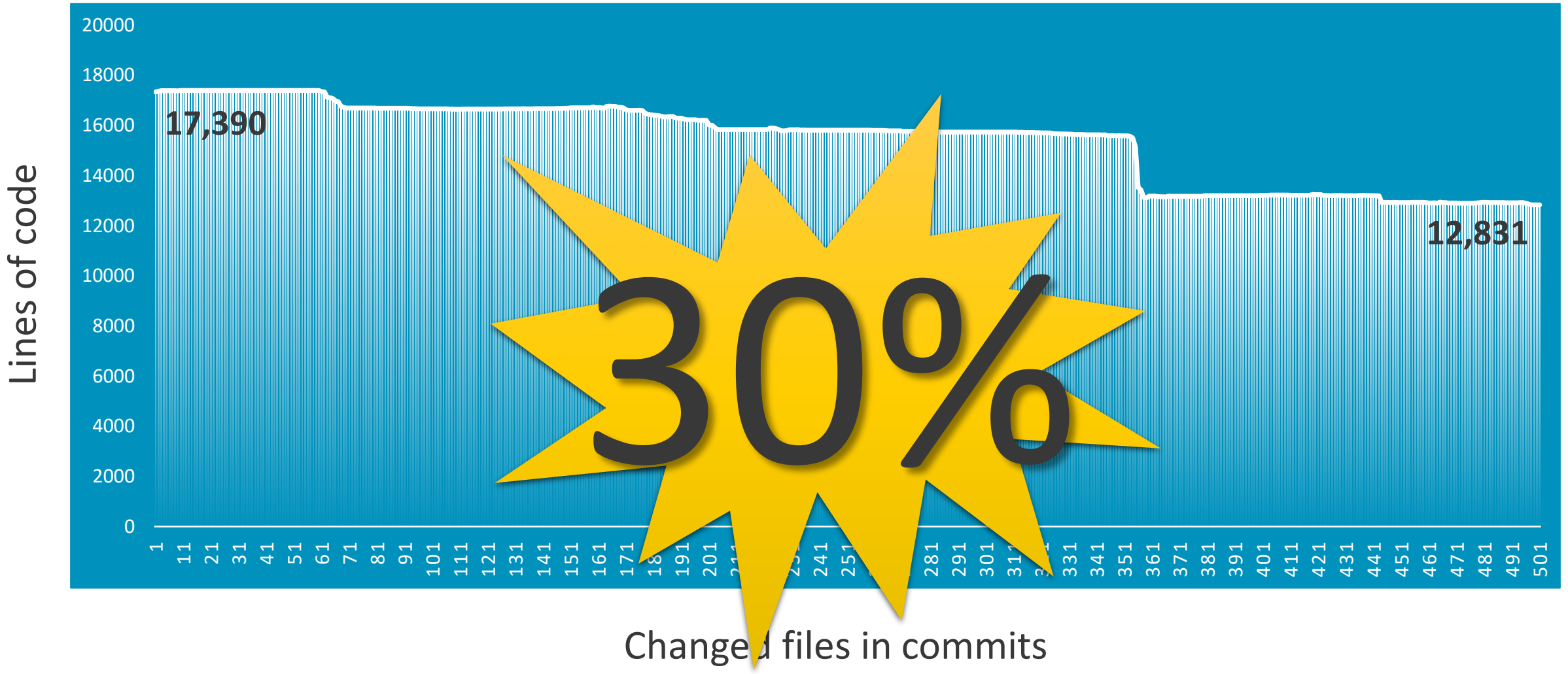
NOTE: For some of the reported defects, checkpatch may be able to
mechanically convert to the typical style using --fix or --fix-inplace.

NOTE: Whitespace errors detected.
You may wish to use scripts/cleanpatch or scripts/cleanfile

ALL PICTURES SHOWN ARE FOR ILLUSTRATION PURPOSE ONLY. ACTUAL HELL MAY VARY.

arm

CCREE driver lines of code count over time



arm

What got fixed?

Reinventing the wheel

Backwards compatibility

Using the wrong API

Duct tape engineering

Zombie code

Macro gymnastics

Don't repeat yourself

arm

Reinventing the wheel

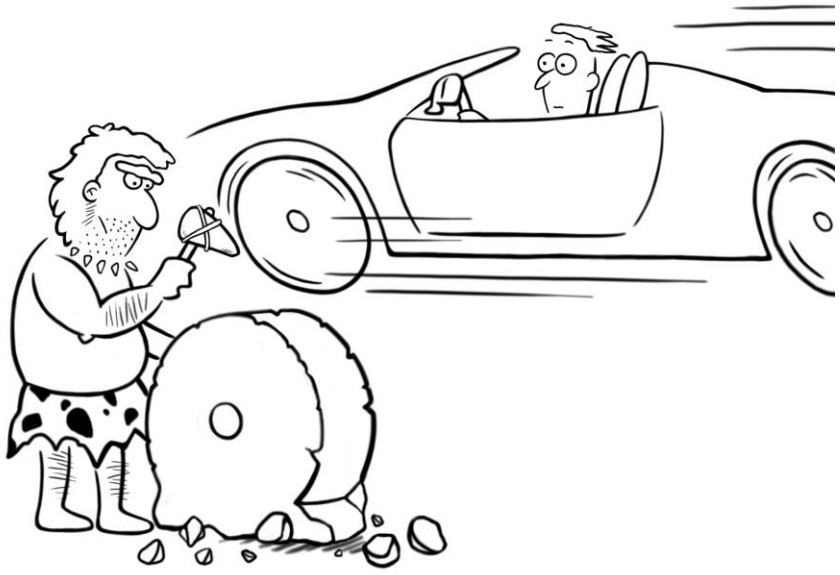
```
void ssi_buffer_mgr_copy_scatterlist_portion(
        u8 *dest, struct scatterlist *sg,
        uint32_t to_skip,  uint32_t end,
        enum ssi_sg_copy_direct direct)
{
        struct scatterlist t_sg;
        struct scatterlist *current_sg = sg;
        int sg_index, cpy_index;
        uint32_t s
        uint32_t lb
        nents = ssi_buffer_mgr_get_sgl_nents(sg, end, &lbytes, NULL);
        sg_index = current_sg->length;
        while (sg_index <= to_skip) {
                current_sg = sg_next(current_sg);
                //if have reached the last (this is the next to last) entry
                if (current_sg == NULL)
                        return;
                sg_index += current_sg->length;
                nents--;
        }
        cpy_index = current_sg->length - sg_index + to_skip;
        t_sg = *current_sg;
        /*update the offset in the sg entry*/
        t_sg.offset += current_sg->length - cpy_index;
        /*copy the data*/
        if (direct == SSI_SG_TO_BUF) {
                sg_copy_to_buffer(&t_sg, 1, dest, cpy_index);
        } else {
                sg_copy_from_buffer(&t_sg, 1, dest, cpy_index);
        }
        current_sg = sg_next(current_sg);
        nents--;

        if (end > sg_index) {
                if (direct == SSI_SG_TO_BUF) {
                        sg_copy_to_buffer(current_sg, nents,
                                        &dest[cpy_index], end - sg_index);
                } else {
                        sg_copy_from_buffer(current_sg, nents,
                                        &dest[cpy_index], end - sg_index);
                }
        }
}
```

```
void cc_copy_sg_portion(struct device *dev, u8 *dest, struct scatterlist *sg,
        u32 to_skip, u32 end, enum cc_sg_cpy_direct direct)

u32 nents, lbytes;

nents = cc_get_sgl_nents(dev, sg, end, &lbytes, NULL);
cc_copy_sg_portion(dest, sg, to_skip,
}
```

What does this code do?

Is this function unique to my module?

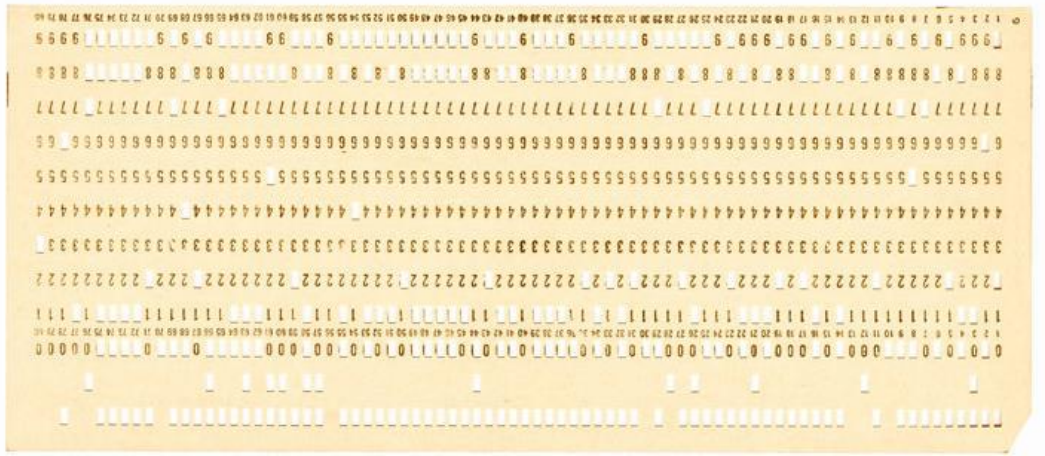
If not, what are the other users doing?



Backwards compatibility

“If you’re backwards compatible, you’re really backwards”

-- Don Mattrick

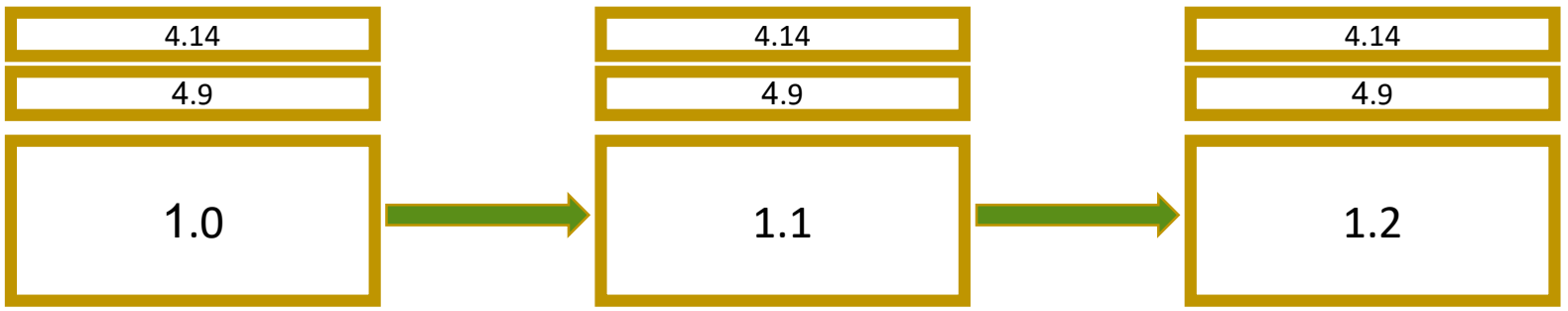


```
void ssi_buffer_mgr_unmap_aead_request(
        struct device *dev, struct aead_request *req)
{
        struct aead_req_ctx *areq_ctx = aead_request_ctx(req);
#if LINUX_VERSION_CODE >= KERNEL_VERSION(4,3,0)
        struct crypto_aead *tfm = crypto_aead_reqtfm(req);
        uint32_t dummy;
        bool chained;
        uint32_t size_to_unmap = 0;
#endif
        if (areq_ctx->mac_buf_dma_addr != 0) {
                SSI_RESTORE_DMA_ADDR_TO_48BIT(areq_ctx->mac_buf_dma_addr);
                dma_unmap_single(dev, areq_ctx->mac_buf_dma_addr,
                        MAX_MAC_SIZE, DMA_BIDIRECTIONAL);
```


arm

Out-of-tree driver kernel and product version management

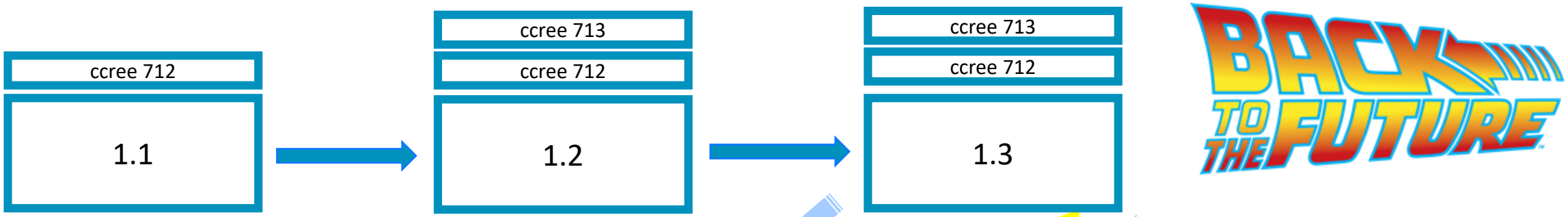
CCREE 712



CCREE 713

arm

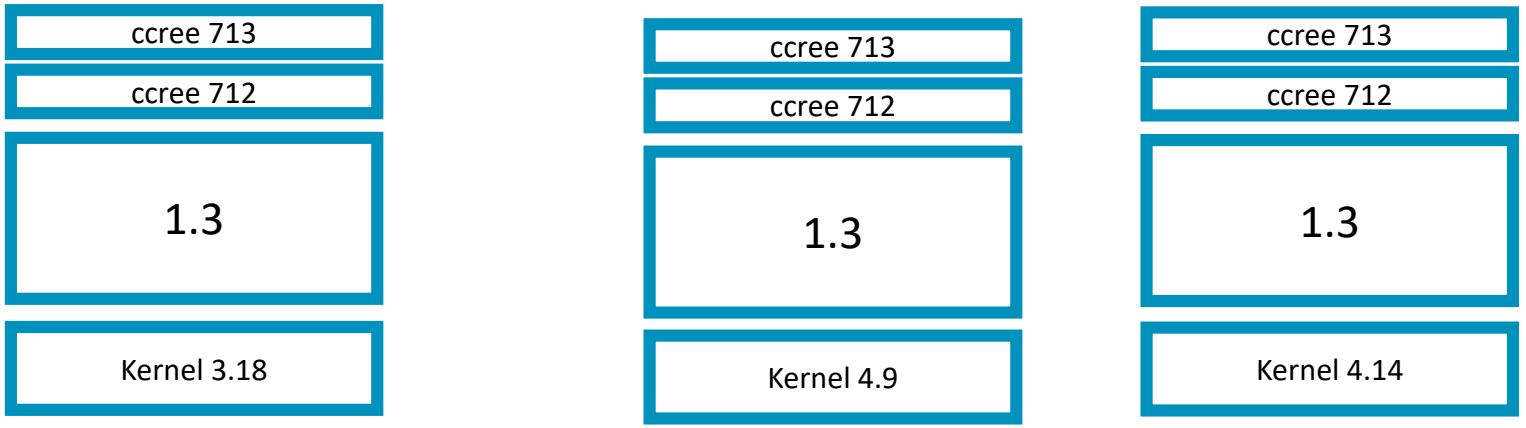
Upstream driver kernel and product version management



Latest kernel

https://backports.wiki.kernel.org

"The *Backports Project* enables old kernels to run the latest drivers."



Stable kernels backports

arm

Using the wrong API

gby@sugar:~/src/cc_sw/.../linux$ wc -l ssi_sysfs.c
436 ssi_sysfs.c

“sysfs - _The_ filesystem for exporting kernel objects.

What it is:
~~~~~~~~~~~~

sysfs is a ram-based filesystem initially based on ramfs. It provides a means to export kernel data structures, their "attributes, and the linkages between them to userspace.”

gby@sugar:~/src/linux/.../ccree$ wc -l cc_debugfs.c
101 cc_debugfs.c

Debugfs exists as a simple way for kernel developers to make information available to user space. Unlike /proc, which is only meant for information about a process, or sysfs, which has strict one-value-per-file rules, debugfs has no rules at all. Developers can put any information they want there.

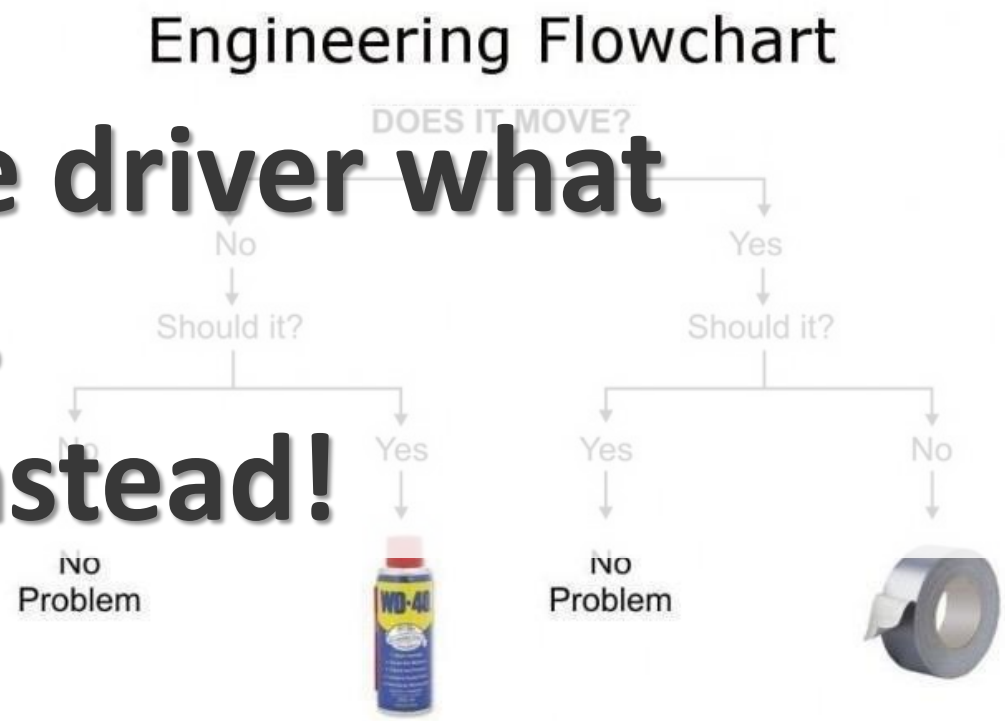
arm
~~~~~~~~~~~~

Duct Tape Engineering

```
From 785979d093200b658f7700ce8a1dbb6543c7675a Mon Sep 17 00:00:00 2001
From: Gilad Ben-Yossef <gilad@benyossef.com>
Date: Sun, 15 Jan 2017 12:28:44 +0200
Subject: [PATCH v3 RESEND] dm: switch dm-verity to async hash crypto API
MIME-Version: 1.0
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 8bit

Use of
CPU bas
algorit
potenti

This ca
boot ti
protect

Move DM_VERITY to the asynchronous hash API.

Signed-
Tested-
CC: Eri
CC: Ond
---

Resending since I did not get any feedback beyond Milan confirmation that
the latest version passes his simple sanity test.

The patch was tested by me on an Armv7 based dual core Zynq ZC706 development
board with SHA256-asm, SHA256-neon synchronous providers with no visible
degradation of performance, with cryptd based asynchronous versions of the
same and with an off tree Arm CryptoCell asynchronous provider.

Changes from v2:

- Use completion to potentially wait also on crypto_ahash_init() as it
  may finish asynchronously as well in some drivers, such as cryptd, as
```

Don’t try to fix in the driver what is broken elsewhere.

Fix the framework instead!



arm

Macro Gymnastics

```
#define DX_HOST_IRR_REG_OFFSET       0xA00UL

#define DX_BASE_HOST_RGF 0x0UL
#define READ_REGISTER(_addr) ioread32((_addr))
#define CC_HAL_READ_REGISTER(offset) READ_REGISTER(cc_base + offset)
#define CC_REG_OFFSET(unit_name, reg_name)          \
      (DX_BASE_ ## unit_name + DX_ ## reg_name ## _REG_OFFSET)
irr = CC_HAL_READ_REGISTER(CC_REG_OFFSET(HOST_RGF, HOST_IRR));
```

```
#define CC_REG(reg_name) CC_ ## reg_name ## _REG_OFFSET
static inline u32 cc_ioread(struct cc_drvdata *drvdata, u32 reg)
{
      return ioread32(drvdata->cc_base + reg);
}
irr = cc_ioread(drvdata, CC_REG(HOST_IRR));
```


arm

Zombie Code

Dead code is buggy code

```
gby@sugar:~/src/linux/drivers/staging/ccree$ git log --oneline . | egrep '(dead|used)'
642ed0c staging: ccree: remove dead code
c5bf891 staging: ccree: remove unused and redundant variable idx
2af630f Staging: ccree: Remove unused variable monitor_lock
707c76a staging: ccree: remove unused completion
17d46da Staging: ccree: Remove unused variable.
e8e5110 staging: ccree: remove unused type CCFipsSyncStatus_t
b091fad staging: ccree: remove unused function
1c0cccd staging: ccree: remove dead code
ef78342 staging: ccree: drop no longer used macro
c928f1d staging: ccree: remove unused struct
841d1d8 staging: ccree: remove unused debug macros
6c5ed91 staging: ccree: remove unused function argument
b4573c9 staging: ccree: remove unused code
```

arm

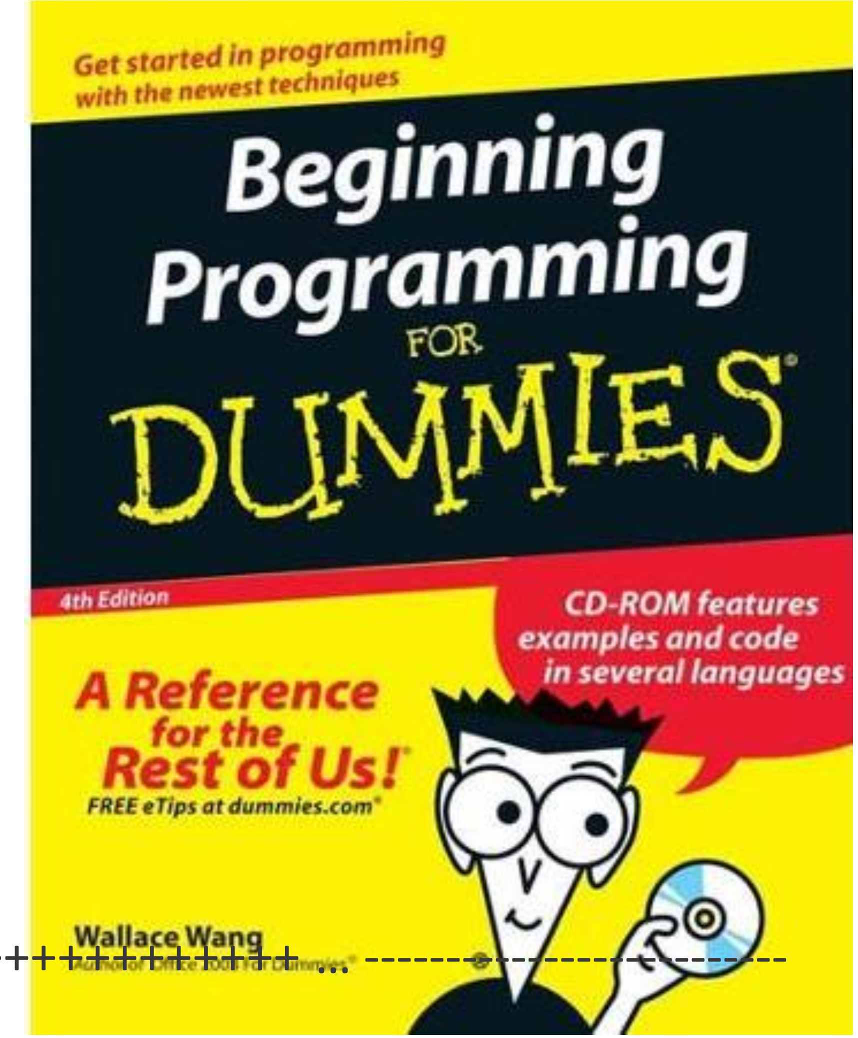
Don’t repeat yourself

commit 28b1ad901fce45c0f3027f7380fa1589d4bfc1b4
Author: Gilad Ben-Yossef <gilad@benyossef.com>
Date: Sun Jan 7 12:14:32 2018 +0000

    staging: ccree: fold common code into service func

    Fold common code in hash call into service functions.

    Signed-off-by: Gilad Ben-Yossef <gilad@benyossef.com>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>



 drivers/staging/ccree/ssi_hash.c | 339 ++ ... ---

 1 file changed, 116 insertions(+), 223 deletions(-)

arm

The Happy End

```
commit 8151da11c5624c8d43ffbc90e93fa1ecdc4da9ab
Author: Gilad Ben-Yossef <gilad@benyossef.com>
Date:   Thu Mar 8 07:39:48 2018 +0000

    staging: ccree: remove ccree staging copy

    Now when the ccree driver has been accepted into
    the cryptodev tree we can remove the staging tree copy.

...
    Signed-off-by: Gilad Ben-Yossef <gilad@benyossef.com>
```

arm

Thank You!
Danke!
Merci!
谢谢!
ありがとう!
Gracias!
Kiitos!

Alexander Mazyrin <algoritmist1618@gmail.com>
Andrew Morton <akpm@linux-foundation.org>
Arnd Bergmann <arnd@arndb.de>
Arvind Yadav <arvind.yadav.cs@gmail.com>
Bincy K Philip <bincy_k_philip@yahoo.co.in>
Branislav Katreniak <brano@ksp.sk>
Colin Ian King <colin.king@canonical.com>
Connor Kelleher <connor.r.kelleher@gmail.com>
Dan Carpenter <dan.carpenter@oracle.com>
David S. Miller <davem@davemloft.net>
Derek Robson <robsonde@gmail.com>
Dhananjay Balan <mail@dbalan.in>
Geert Uytterhoeven <geert@linux-m68k.org>
Gennadii Altukhov <grinrag@gmail.com>
Greg Kroah-Hartman <gregkh@linuxfoundation.org>
Gustavo A. R. Silva <garsilva@embeddedor.com>
Herbert Xu <herbert@gondor.apana.org.au>
Ian Chard <ian@chard.org>
Jeremy Sowden <jeremy@azazel.net>
Jhih-Ming Hunag <fbihjmeric@gmail.com>
Kamal Heib <kamalheib1@gmail.com>
Karthik Tummala <karthik@techveda.org>
Logan Gunthorpe <logang@deltatee.com>
Matthew Giassa <matthew@giassa.net>
Pravin Shedge <pravin.shedge4linux@gmail.com>
Pushkar Jambhlekar <pushkar.iit@gmail.com>
Raphaël Beamonte <raphael.beamonte@gmail.com>
Rishabh Hardas <rishabhhardas@gmail.com>

arm

Simon Sandström <simon@nikanor.nu>
Srishti Sharma <srishtishar@gmail.com>
Stephen Brennan <stephen@brennan.io>
Suniel Mahesh <sunil.m@techveda.org>
Timothée Isnard <timotheecisnard@gmail.com>
Tyler Olivieri <sleepingzucchini@gmail.com>
Wei Yongjun <weiyongjun1@huawei.com>

Questions?

NO FISH WERE HARMED DURING
THE MAKING OF THIS PRESENTATION.

arm